Density-Aware Sliders

Paymahn Moghadasian
University of Manitoba
66 Chancellors Cir
umpaymah@cs.umanitoba.ca
?

Pourang Irani
University of Manitoba
66 Chancellors Cir
irani@cs.umanitoba.ca
?

Niklas Elmqvist
Purdue University
?
elm@purdue.edu
?

ABSTRACT
In this paper we present a selection tool that utilizes pre-existing knowledge of the distribution of the underlying data to ease browsing and selection. Standard models of the slider selection tool suffer from two problems: poor subpixel data querying, and uniform visual representation of non-uniform data. We provide density-aware interactions to give users more efficient subpixel data querying and use embedded visualisations to better represent the data encoded by the tools. Through several controlled user studies, we find that our proposed density-aware slider outperforms standard Alphasliders.

Author Keywords
Dynamic query, information visualization, slider, data selection, Alphaslider, lasso, range slider

ACM Classification Keywords
Need help with this

General Terms
Need help with this

INTRODUCTION
Dynamic queries provide easy to use, powerful and efficient tools and interfaces which allow users to rapidly and reversibly query data and uncover trends in the data being explored [1]. Currently, there are many tools available to users which benefit from the advantages of dynamic queries such as the slider, checkbox or button. The Alphaslider is a slider designed to query large lists of alphabetically sorted alphanumeric items [2].

The Alphaslider suffers from two problems. 1) The Alphaslider does not have an awareness of the data it encodes. This is especially important when the interfaces map more than one item to any pixel. Subpixel querying is a new problem because information density is ever increasing. With highly dense data more items are mapped to each pixel making efficient data selection increasingly difficult. 2) The Alphaslider gives users very poor visual feedback regarding general density and the users "location" during subpixel querying.

CHI'12, May 5–10, 2012, Austin, Texas, USA.

We propose density-aware sliders. Our sliders are knowledgeable of the data they encode and use this information to simplify data querying. Our most efficient slider interface intelligently redistributes the items in densely packed pixels over a larger area and provides users with a list of nearby items giving the user a sense of location within the pixel and quick access to each item. To solve the issue of data visualisation we used embedded visualisations such as those proposed by Willett et al [3]. These visualizations aid in estimation of the density of items in a given pixel.

Our results show that users can query significantly faster with density-aware sliders when compared against the standard tools. Our ActiveList Slider allowed subjects to query significantly faster than the Alphaslider. Should I put in some numbers here?

This paper provides novel solutions to an aging problem and demonstrates that intelligent selection tools have the potential to improve upon the status quo. Although the proposed solutions perform more efficiently than previous standards there is still work to do in fine-tuning and possibly developing hybrid designs that take advantage of matured and novel ideas. In this paper we discuss will present the problem in greater detail, discuss related works and continue to describe the experimental design and results of our controlled user study. We will conclude the paper by discussing the implications of the research and potential future work.

RELATED WORK
A significant amount of research has been done exploring dynamic query interfaces. Studies have shown that dynamic queries allow users to interactively and rapidly form queries [1] to explore datasets and discover trends [4]. FilmFinder by Ahlberg et al. [5] allowed users to search a database of movies through the use of several dynamic query tools. Users could explore the data to find that, for example, Western movies were more prevalent in the 1970s than in the 2000s. The Dynamic HomeFinder, an idea proposed by Williamson et al. allowed users to explore a real estate market to find homes satisfying their needs [6]. For example, the Dynamic HomeFinder allowed user to see how changing their acceptable price range would affect the available houses and their properties. The Dynamap [7]

allowed users to explore trends in health statistics by overlaying health information on a map of the United States of America. Tools, such as the slider, were available to query different parameters to in order to explore the data. The paradigm we have built our experiments around is very similar to that of the FilmFinder where users can interact with and explore a database of movies. Users are given several dynamic query widgets which can be used to filter the movies and discover trends.

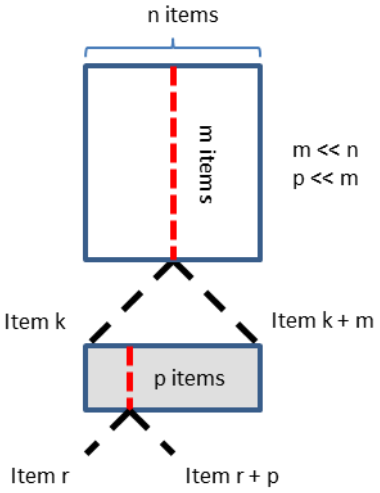
Standard dynamic query interfaces such as the slider, checkbox or button offer sufficient functionality in many cases; however, there have been many proposed improvements to these traditional interfaces. The Alphaslider [2] is a slider which allows users to quickly select a single item from an alphanumerically sorted list. The slider offers coarse and fine movement through a two tiled thumb and arrow buttons. The TrapezoidBox is a dynamic query tool which allows users to specify spatial proximity queries [8]. The TrapezoidBox allows users to specify queries such as: if the restaurant is within 300 meters then it can be 3 stars or higher; if the restaurant is with 500 meters then it must be rated 4 stars or higher; if the restaurant farther than 500 but no further than 1 kilometer then the restaurant must be rated 5 stars. Lanning et al [9] offer a novel way of interacting with and visualizing multidimensional data called MultiNav. The visual metaphor best associated with MultiNav is that of sliding rods where each rod (or slider) is linked to a dimension of data. Users can select value ranges along each rod to interact with the data. The Alphaslider is the baseline slider for our experiment. The Alphaslider was chosen for two purposes, namely its efficiency and versatility. MultiNav was not an inspiration for our research because it is not suited to precise data selection in large data sets.

Very often there is a need for highly precise parameter selection. These situations arise in real-world scenarios where the range of parameters vastly exceeds the available number of pixels. For example, an hour long movie filmed at 24 frames per second has 86 400 frames. Using a slider to access individual frames for editing can be tedious. The Alphaslider [2] was designed with situations such as this in mind. The PVSlider [10] and FineSlider [11] both aim to improve upon the Alphaslider. The PVSlider uses a popup vernier to give users sub-pixel-pitch control and sub-pixel visualization. The FineSlider uses an elastic band metaphor to give users more intuitive control over scrolling speed and precision where a longer elastic band correlates to faster scrolling and a shorter band correlates to more precise and deliberate item selection. The Zlider, proposed by Ramos et al, takes advantage of pressure input in a pressure sensitive environment to fluidly change from coarse to fine granularity and shift granularity control from the system to the user [12]. Aspects of the PVSlider and FineSlider overlap with the purpose of this research but they were not given priority over the Alphaslider for several purposes: 1)The FineSlider, as shown in the experiment by Ayatsuka et al in [10] performed worse than the Alphaslider and was less favoured by the participants. 2) The PVSlider, although marginally faster, gives subpixel visual feedback which is reliant on monodensity information across the slider which is not required by the Alphaslider. The Zlider was not chosen because there is no way for users to give pressure input to our system.

DESIGNING OF MULTI-ITEM PIXEL ASSIGNMENT

Mapping multiple items to a single pixel and giving users quick access to each item remains an ever-present problem. The Alphaslider gets around this issue by giving users list based movement where the user navigates the list of items encoded by the slider through coarse or fine grained dragging. Coarse dragging lets the user skip through the list 10 items at a time while fine grained dragging lets the user move through the list 1 item at a time. This causes a problem to arise when multiple lists are present of different densities; a slider encoding 10 000 items will feel heavier than a list encoding 1 000 items causing difficulty for the user in predicting how quickly they need to drag the slider. The Alphaslider's querying method gives no visual feedback when the items per pixel exceeds the coarse grained movement value making highly dense data even more confusing.

Subpixel visual feedback is also an issue intricately associated with the Alphaslider. This problem was identified by Ayatsuka et al and they attempted to rectify it. However, their solution lacks the ability to dynamically adapt to the encoded data. The popup vernier in the PVSlider requires that the developer be explicitly aware of the information to give the slider appropriate scales for the popup vernier.



Approaching this problem with multi-level querying in mind can solve the above issues. By giving the user multi-level querying the issue of slider "weight" is solved because the slider's movement is pixel based instead of list based. Items per pixel becomes a non-issue in terms of visual feedback because coarse querying is pixel based instead of item based. Subpixel querying is partially solved because items mapped in a single pixel are instead mapped along the width of slider which can be individually selected. This solution requires refinement. We will present our final solution later in the paper.

EXPERIMENTAL DESIGN

Introduction

An experiment was conducted to compare different designs of density aware sliders against the Alphaslider.

Apparatus

The interfaces used in the experiment were built using Visual Studio 2010 Professional. A 23.5 inch Dell monitor with a resolution of 1920x1080 pixels with a standard 3 button mouse was used. Query results were displayed in Microsoft Sans Serif with a font size of 8.25 while query targets were displayed in Microsoft Sans Serif with a font size of twelve. An Intel i5-2400 CPU with a clock speed of 3.1 GHz was used along with 4 GB of RAM.

Interfaces

A total of 3 different sliders were used in the experiment (Figures x through x+2).

Alphaslider

The Alphaslider (Figure 2) has four navigation techniques. Users can jump directly to an item in the list by clicking anywhere in the bounded area above the letters. Users can navigate through the list at a rate of ten items per mouse movement by clicking in the top tile of the slider thumb and dragging. Users can also navigate one item at a time either by clicking on the arrows at the ends of the slider or by clicking in the bottom tile of the slider thumb and dragging.

ActiveArea Slider

The ActiveArea Slider (Figure 3) allows users to navigate by clicking on and dragging the slider thumb, by clicking on and dragging the secondary red slider or by pressing the left or right arrow keys on the keyboard. The main slider has a variable size which changes based on the density of information of the pixel it queries mapping a maximum of 2 items for each horizontal pixel of the slider. Users can drag the red triangle to query items mapped to the main slider. Users can also roll the mouse wheel to navigate 1 item at a time. Rolling the mouse wheel appropriately moves the secondary slider and gives users continuous querying whereas dragging the red slider is bound by the edges of the main slider.

ActiveList Slider

The ActiveList Slider (Figure 5) is a hybrid design of the ActiveArea Slider and the List slider. The list in this slider is based on the subpixel item being queried. The items of the list are the items r through r + p. If p is less than a minimum threshold then the list is enumerated up until that threshold is met or until item k + m is shown. Dragging the red slider and rolling the mouse wheel appropriately update the list. Dragging is bound by the edges of the slider while using the mouse wheel gives users continuous querying.

Hypotheses

This paper is primarily concerned with designing the most efficient slider. While accuracy is an important factor in designing a slider, that attribute falls largely upon the user. Because of this speed is the most significant factor to measure. The speed with which a user locates an item is largely dependent on total mouse movement. For each of the sliders there is a period of querying where the user approximates the area of the target and a period of querying where the user does fine adjustments to acquire the target. Based on this assumption the following hypotheses can be made:

1) The ActiveList slider will perform the best because it does the best job breaking the query into multiple levels.
2) All of our sliders will outperform the Alphaslider because they give the user better control and visual feedback.
3) Error rates among sliders will be similar.
4) The ActiveList Slider will be preferred because the technique gives users better, quicker and easier access to the data.

Experiment Variables

Independent Variables

- Type of interface (*AlphaSlider*, *ActiveArea Slider* or *ActiveList Slider*)
- Distortion Style (*Input* or *Display* distortion)
- Data size (*Small*, *Medium* or *Large*)

Dependent Variables

i) Speed of acquisition

ii) Error rate

iii) Subjective satisfaction

Tasks

A total of 12 subjects completed four trials for each condition. Subject age ranged from 20 to 39. Each target was randomly generated at runtime. With each new task the thumb was returned to the beginning of the slider.

Procedures

A pilot study was first done to weed out poorly designed and inefficient interfaces. Prior to the timed trials the subjects were given a set of tasks under each condition (the ordering of which was determined by a balanced Latin square) minutes to familiarize themselves with each interface while reading interface specific instructions and ask questions. During the timed trial the subjects were not allowed to ask questions and were asked to fill out a NASA TLX form for each interface and distortion style. Upon completion of the experiment the subjects were given a forced pairwise comparison sheet on which they indicated their preferred interface for every combination of technique and distortion style. Subjects were also asked to write any comments they have regarding the experiment, interfaces and distortion styles.

RESULTS

Acquisition Time: We found no statistical difference between techniques ($F(2) = 2.040$, $p = 0.154$) upon removal of outliers. Upon exclusion of the smallest data size our significance rises ($F(2) = 2.829$, $p = 0.081$) which leads us to believe that larger data sizes benefit our data aware sliders. There was a statistically significant difference between distortion styles ($F(1) = 8.153$, $p < 0.02$).The Alphalslider was on average 1.1 seconds slower than our interfaces regardless of distortion style. This supports hypothesis 2 but not hypothesis 1.

Error Rate: We found no significant difference in error rate among techniques ($p = 0.712$) or among distortion styles ($p = 0.316$). Subjects were 99% accurate with all interfaces and distortion styles. This validates hypothesis 3.

TLX Analysis: Analysis of the TLX information indicates that there were significant main effects for frustration and effort among interfaces ($X^2(2) = 7.386$, $p < 0.05$ and $X^2(1) = 18.286$, $p < 0.001$, respectively). No significant main effects were found for distortion style for either frustration or effort ($X^2(2) = 0.034$, $p = 0.853$ and $X^2(1) = 0.533$, $p = 0.465$, respectively). Significant interaction effects were found for effort ($X^2(5) = 22.134$, $p < 0.001$) and for frustration ($X^2(5) = 12.481$, $p < 0.03$). Post-hoc analysis with Wilcoxon Signed-Rank Tests and Bonferroni corrections revealed statistically significant differences in effort for the following pairs(Input Distortion will be reffered to as “ID” and Display Distortion will be reffered to as “DD”): ID-ActiveList Slider vs ID-Alphaslider ($Z = -2.940$, $p = 0.003$), DD-ActiveList Slider vs ID-Alphaslider ($Z = -2.809$, $p = 0.005$), ID-ActiveList Slider vs ID-ActiveArea Slider ($Z = -2.634$, $p = 0.008$), DD-ActiveList Slider vs ID-ActiveArea Slider ($Z = -2.849$, $p = 0.004$). The post-hoc analysis also revealed no significant differences for frustration. This information supports hypothesis 4.

Subjective Preference: Dr. Irani asked me to analyze this information later

DISCUSSION

Our results show small but statistically insignificant improvements in target acquisition times. Both the ActiveArea Slider and ActiveList Slider perform 1.1 seconds (7.3%) faster than the Alphaslider. There was an insignificant difference in error rate between interfaces. When comparing the interfaces with the smallest data set removed we see a jump in statistical significant, from $p = 0.154$ to $p = 0.081$ which suggests that a larger dataset might benefit more from our new sliders. We ran a small pilot (5 participants) after our experiment with a dataset of approximately 50 00 items, 2.5 times larger than the largest dataset of the experiment. A post-hoc analysis of the data collected from the pilot shows statistically significant differences between the ActiveList Slider and ActiveArea Slider ($p < 0.001$) and the ActiveList Slider and Alphaslider ($p = 0.025$). This gives reason to believe that the ActiveList Slider becomes truly benefcial in highly dense data.

Overall, subjects preferred our sliders over the Alphaslider. This is because the Alphaslider requires that users have fine motor skills to query 1 item at a time or to click on arrow buttons repeatedly while our interfaces let users use the mousewheel which is highly precise and requires less motor skills without sacrificing speed. Many users commented, “I really like using the mousewheel” and even sometimes instinctively attempted using the mousewheel while querying with the Alphaslider.

We suspect that with ultra-dense datasets (greater 100 000 items) we will start seeing performance degradation with our sliders because of the instinctive behavior of users. Ideally users should use the different graunarities provided by our sliders: the main thumb, the secondary thumb and the mousewheel. Users tend to not use the secondary thumb because it is perceived as having the same granularity as the mousewheel, which is true for small to medium density datasets but not for larger datasets. Their granularity starts to deviate noticeably at approximately 20 000 items. We believe that this issue can be avoided with explicit training or possibly with a new visual cue.

CONCLUSIONS

AKNOWLEDGEMENTS

We thank NSERC for funding part of this research and the participants for the valuable time and effort.

REFERENCES

[1] Shneiderman B., "Dynamic Queries for Visual Information Seeking," Maryland, 1994.

[2] Christopher Ahlberg and Ben Shneiderman, "The Alphaslider: A Compact and Rapid Selector," in *Human Factors in Computing Systems*, Boston, 1994, pp. 365-371.

[3] Wesley Willett, Jeffrey Heer, and Maneesh Agrawala, "Scented Widgets: Improving Navigation Cues with Embedded Visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1129-1136, November 2007.

[4] Christopher Ahlber, Christopher Williamson, and Ben Shneiderman, "Dynamic Queries for Information Exploration: An Implementation and Evaluation," in *CHI*, 1992, pp. 619-626.

[5] Christopher Ahlberg and Ben Shneiderman, "Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays," in *CHI*, 1994, pp. 313-317.

[6] Christopher Williamson and Ben Shneiderman, "The Dynamic HomeFinder: Evaluating Dynamic Queries in a Real-Estate Information Exploration System," in *ACM SIGIR*, 1992, pp. 338-346.

[7] Catherine Plaisant and Vinit Jain, "Dynamaps: dynamic queries on a health statistics atlas," in *CHI Conference Companion on Human Factors in Computing Systems*, 1994, pp. 439-440.

[8] Myoungsu Cho, Bohyoung Kim, Dong K Jeong, Yeong-Gil Shin, and Jinwook Seo, "Dynamic Query Interface for Spatial Proximity Query with Degree-of-Interest Varied by Distance to Query Point," in *CHI*, 2010, pp. 693-702.

[9] Tom Lanning, Kent Wittenburg, Michael Heinrichs, Christina Fyock, and Glenn Li, "Multidimensional Information Visualization through Sliding Rods," in *AVI*, 2000, pp. 173-180.

[10] Yuji Ayatsuka, Jun Rekimoto, and Satoshi Matsuoka, "Popup Vernier: a Tool for Sub-pixel-pitch Dragging with Smooth Mode Transition," in *11th Annual ACM Symposium on User Interface Software and Technology*, 1998, pp. 39-48.

[11] Toshiyuki Masui, Kouichi Kashiwagi, and George R. IV Borden, "Elastic Graphical Interfaces to Precise Data Manipulation," in *CHI Conference Companion on Human Factors in Computing Systems*, 1995, pp. 143-144.

[12] Gonzalo Ramos and Ravin Balakrishnan, "Zliding: Fluid Zooming and Sliding for High Precision Parameter Manipulation," in *18th Annual ACM Symposium on User Interface Software and Technology*, 2005, pp. 143-152.

The columns on the last page should be of approximately equal length.
Remove these two lines from your final version.